REMARKS

This Amendment is filed response to the non-final Office action dated March 31, 2004. All objections and rejections are respectfully traversed. Reconsideration of the application, as amended, is respectfully requested.

It appears that the Information Disclosure Statement (IDS), dated February 13, 2004, was not considered in the non-final Office action mailed March 31, 2004. Accordingly, the Applicants respectfully request that the references cited in their IDS of February 13, 2004 be considered in the Patent Office's next correspondence. Another copy of the IDS is included herewith.

Claims 1-23 are in the case. The Applicants have added new claims 13-23 which include the same or similar subject matter as originally filed claims 1-12. Notably, claims 13-14 and 20 comprise the same or similar patentable subject matter identified in the "Allowable Subject Matter" section of the Office Action, i.e., at paragraph 9, with regards to claims 6 and 12. No new matter is being introduced.

At paragraphs 1-4 in the Office action, claims 1-12 were rejected under the judicially created doctrine of obviousness-type double patenting as being unpatentable over claims 1 and 7 of U.S. Patent No. 6,425,121 to Phillips (hereinafter "Phillips") in view of U.S. Patent No. 5,732,192 to Malin et al. (hereinafter "Malin").

In response, the Applicants submit a Terminal Disclaimer herewith, thereby obviating these double patenting rejections.

At paragraphs 5-6 in the Office action, claims 1, 2, 7 and 8 were rejected under 35 U.S.C. §102(b) as being anticipated by U.S. Patent No. 5,710,896 to Seidl (hereinafter "Seidl").

At paragraphs 7-8 in the Office action, claims 3-5 and 9-11 were rejected under 35 U.S.C. 103(a) as being obvious over Seidl in view of U.S. Patent No. 5,437,029 to Sinha (hereinafter "Sinha").

The present invention, as set forth in representative claim 1, comprises in part:

> A storage medium containing program instructions readable by a computer for detecting and resolving circular flow paths disposed within a flow diagram representing the logical operation of a corresponding application program, the flow diagram formed by interconnecting a plurality of symbolic representations of program objects, the program objects configured to execute associated functions in response to corresponding triggering events, the readable program instructions comprising program instructions for:
>
> ***establishing a busy indicator at a given program object, the busy indicator signifying whether the given program object is currently <u>executing</u> its associated function***;
>
> ***in response to the occurrence of the given program object's triggering event, testing the respective busy indicator***;
>
> ***if the busy indicator signifies that the given program object is currently executing, blocking the given program object from re-executing in response to the triggering event;***
>
> ***if the busy indicator signifies that the given program object is not currently executing, permitting the given program object to execute in response to the triggering event.***

Fig. 6 in Seidl illustrates an exemplary arrangement of graphical objects, including a TEcho unit 600 and a speaker unit 620. An application developer can create logical connections between different "ports" on the graphical objects. For instance, as shown in Figs. 6 and 7, a connection 610 is made by drawing a line from an output port on the

TEcho unit 600 and connecting the line to an input port on the speaker unit 620. See Seidl, Col. 10, lines 46-53. Seidl discloses a topology check that ensures that a "circular path" is not created by connecting both endpoints of the line to the same input port. See Seidl, Col. 11, lines 9-16.

The preferred embodiment in Seidl discloses a "semantic snapping" technique that provides visual feedback when a connection has been established between the two graphical objects. More specifically, if the application developer drags his/her cursor within a predetermined "snapping radius" of an object's port, then the logical connection becomes visually "snapped" (i.e., connected) to that port. See Seidl, Col. 10, lines 59-67.

The Applicants respectfully urge that Seidl does not teach or otherwise suggest ***establishing a busy indicator at a given program object, the busy indicator signifying whether the given program object is currently executing its associated function***, as recited in Applicants' claim 1.

At the top of page 4 in the Office action, it appears that the Applicants' claimed "busy indicator" is equated with the "snap-enter event" disclosed in Col. 11, lines 12-13 in Seidl. The Applicants respectfully disagree with this equivalence. The snap-enter event in Seidl occurs when an endpoint of a graphical line is connected to a port of a graphical object ("on the snap-enter event, the connection's endpoint once again has locked on to an input port"). Thus, Seidl's snap-enter event occurs at application-development time as an application programmer graphically interconnects a plurality of graphical objects.

In contrast, the claimed busy indicator does not concern graphical connections made while an application is being developed. Instead, the busy indicator is a run-time indicator that signifies whether a given program object is currently *executing* in a flow diagram. Thus, while Seidl's snap-enter event is only relevant at program-development time for providing visual feedback to an application developer with regards to connecting graphical objects, the Applicants' busy indicator functions at run time for signifying whether a given program object is currently *executing* its associated function.

Because Seidl's snap-enter event is not a run-time event, it cannot signify whether a given program object is currently *executing* its associated function, as recited in Applicants' claim 1. For this reason, the Applicants further submit that Seidl also cannot teach or disclose the following run-time events recited in Applicants' claim 1:

> ***in response to the occurrence of the given program object's triggering event, testing the respective busy indicator***;
> ***if the busy indicator signifies that the given program object is currently executing, blocking the given program object from re-executing in response to the triggering event;***
> ***if the busy indicator signifies that the given program object is not currently executing, permitting the given program object to execute in response to the triggering event.***

For the foregoing reasons, the Applicants respectfully submit that independent claim 1 is allowable over the cited art. Applicants further submit that independent claims 7, 13-15 and 21-23 contain the same patentable subject matter as claim 1, and are therefore allowable for the same reasons. The Applicants respectfully submit that claims 2-6, 8-12 and 16-20 depend on allowable claims 1, 7 and 15 and are also allowable for the reasons set forth above.

Accordingly, the Applicants submit that all claims are in condition for allowance and early favorable action is requested.

Please charge any additional fee occasioned by this paper to our Deposit Account No. 03-1237.

Respectfully submitted,

Stephen E. Kabakoff
Reg. No. 51,276
CESARI AND MCKENNA, LLP
88 Black Falcon Avenue
Boston, MA 02210-2414
(617) 951-2500